

SyncWare News

The journal for electronic
and other technical
applications of T/S
computers

Volume 2 Number 1

Sept.-Oct. '84

In case you hadn't noticed, a few changes have occurred with SWN in the last couple of months. The appearance among others, is quite different. Another change that has transpired is the editor/publisher personnel. We are very happy to have Tom Woods as our publisher. Not only is he involved in publishing, but also in subscription building and general accounting as well as moral support. All subscription information, etc., should be addressed to him (see back cover).

Being relatively new at the editing, paste up and necessary general clerical responsibilities, I invite any criticism (hopefully constructive) that you may have about content, appearance, mistakes, etc. Please send your letters, submissions, etc., to me, Tom Bent, and take note of my new address on the back cover.

As for Fred Nachbaur, our Technical Director is taking a well deserved break (albeit a working holiday), and should be back in L.A. for a short while after September 15. Expect an upstate Washington P.O. box number by next issue. His mail is forwarded to me presently. I will send any mail on to him right away.

PLEASE WRITE, to us and specific authors about questions, tips, errors, or to say hello.

Why? Every one involved with SWN is working for zip. Those new subscribers, who do not yet know of SWN's track record, need not fear that their subscription money has gone the way of other TS user mags. ("We all know what TS stands for"). Yes, we read ALL comments, although we can't answer all of them directly. We are interested in keeping communications and resource channels open for you. We need your response and support. Send letters to the authors directly, or send your letters to us. We will take note and forward them ASAP to the proper people.

Speaking of response, your response to the questionnaires has been overwhelming. Thank you. We received close to 100 percent response. When the results are tabulated, we will let you know the outcome. By far, the number one choice for topics was advanced hardware (has anyone created any lately?). Well, as it turns out, we know someone who has. John Oliger has a multiparter that will keep you humming right along. Ray Kingsley has a multiparter on using the 2068 Rom efficiently and properly, in machine code. John Wentworth has a multiparter for 1000 machine code fledglings (like myself). Gary Smith has an unfolding Forth tale, which promises to continue. Fred Nachbaur (the ever present) has a neat application and a way in general to make competing programs agreeable.

We simply don't have room for everything (another change from last year), and must postpone some other excellent material until next issue. We promise more applications, utilities, control, bountiful product reviews, BASIC programming, 3D graphics and more hardware and software. What else can I say?

FOR YOUR SUPPORT

In this column, we will announce any software or hardware that is new or otherwise untested by us. If you have something of interest, we will announce it here. Send us a description of your product. Keep it short, please.

We advise readers to send a SASE (self addressed stamped envelope), as a courtesy, to the individuals or companies to get further information.

IMRE AUERSBAUCHER, 41 King St. A2, Belleville, NJ. 07109, has 4 new programs for ALL TS machines. MATH PACK 1, MATH PACK 2, WEATHERCASTER/ANALYSER, ASTROLOGY. All are \$14.95 pp. per tape. Includes instructions and examples.

GARY SMITH, Hawg Wild Software, PO Box 7668, Little Rock, Arkansas 72217, has quite a bit for your machines. Write to him and tell him what you want. 2068 fig-FORTH (full) and CP SPECTRUM FORTH \$29.95 + \$2.00 p/h. 'EMU-1' SPECTRUM emulator for the 2068, \$60.00 + \$2.00 p/h, Andre Jackson's USER DESIGN GRAPHICS toolkit for the 2068.

DALE LIPINSKI SOFTWARE, 2737 Susquehanna Rd., Roslyn, Pa. 19001, has 5 new programs for the 2068, and 8 new ones for the 1000 and 1500, dealing with accounting and file management. All prices between \$10 and \$20. Dale expresses a commitment to continue providing products. Write for free catalog.

RAY KINGSLEY, Sinware, Box 8032, Santa Fe, NM. 87504, has an update for HotZ 2068. The update eliminates old bugs and has a complete ROM name file (including the EXROM) and description. \$17.00 for the update. \$15.00 for just the documentation. \$40.00 for HotZ 2068.

GAMES TO LEARN BY, PO. BOX 78, Collinsville, CT. 06022, 1-203-673-7089, has obtained a large stock of Timex programs, computers and 2040 printers. Programs include Vu-file, Vu-calc, Vu-3D and FROGGER, to name a few. GTLB also has a SPECTRUM emulator for \$55.00 + \$1.00 p/h. Write for more information.

JASON OLASKY, 8750 Georgia Ave., Silver Spring, Md. 20910, 1-301-588-0537, has a fig-FORTH (full) with editor and assembler available for \$19.95 pp.

A.F.R. Software, 1605 Pennsylvania Ave., 204, Miami Beach, Fla. 33139, 1-305-531-6464, has 3 new programs for 1000's. A "ZX-81 TEXT PROCESSOR" 16-64K. \$10.00. "ZX-CALC" needs 32+K ram. \$11.95. "ZX-81 appointment calender" is available for \$10.00 also. A. RODRIGUEZ will sell ZX81 TIC-TAC-TOE" (16K) for \$5.00 with the purchase of any one of the above programs.

COTTAGE TECHNOLOGY, 5720 W. Little York, Suite 178, Houston, TX. 77091, has ACZ GENERAL LEDGER 2.000 for the 2068. It interfaces with the Cardco numeric keypad (in the joystick port) or the keyboard, and provides a variety of reports on the 2040 printer, 800 entries/mo. up to 150 accounts. \$39.95 + \$1.50 p/h. Includes instruction manual.

GANHART EARTHINGS, 115 N. Rocky River Drive, Berea, Ohio 44017, has a sale on their TYPEX/81 keyboard overlay. It is an adhesive backed die cut vinyl which sticks to the top of the keyboard. Holes keep your fingers in place, providing a positive feel. \$2.50 + \$1.00 p/h.

DEREK STUBBS, Yagsee Publications, PO Box 155, Vicksburg, MI. 49097, former publishers of TS USER newsletter, is cleaning out their stock. They have a complete set of all back issues and a battery power supply called ZEE-PAC, all for \$15.95.

MAXSOFT, Division of MAXSON ELECTRONICS, 611 Franklin St., Hamilton, Ohio 45013, has SUPER SCREEN ROUTINE, a programmers utility that speeds CLS and SCROLL commands and allows for partial screen control (1000, 1500). They also have 3 programs for ALL TS computers, MATH-MASTER, SCRAMBLE BUG and BEAT-THE-SYSTEM, all on one cassette for \$12.95. Specify your computer

WMJ DATA SYSTEMS, 4 Butterfly Drive, Hauppauge, NY. 11788, has a host of machine code spreadsheet programs for ALL TS computers. TAX RETURN ORGANIZER (\$18.00), CHECKREC, STOCK WATCH, APPOINTMENT WATCH, ADDRESS BOOK and INVENTORY (all \$10.00 ea.). 2068 add \$2.00. Discounts to user groups. Write for more information.

KNIGHTED COMPUTERS, 707 Highland St., Fulton, NY. 13069, 1-315-593-8219, has a host of hardware and software from various source. Write or call for catalog and more information.

FORUM

This column is designated for small talk. Forum will be a channel to communicate your needs, ideas and/or specialties to other ZX/TS computerists. Do you provide a service, or do you require a special service? Let us know and we will share your notions with everyone else.

There is one catch, NO PRICES will be announced. This is a first come, first served column, subject to available room. As our advertising picks up, I expect to see this column grow. Those interested should contact the individual directly. If you have trouble, let us know. To get on this band wagon, give us some info. SAS (short and sweet). We'll write it.

BILL FERREBEE, 115 N. 7th. Ave., Paden City, WV. 26159, operates the River Cities Smart BBS at 1-304-652-1416. Modem callers can get in on the Timex interest group to exchange messages, Timex info., product reviews and programs.

RAY RASH, 2424 S.W. 78th. St., Oklahoma City, OK. 73159, 1-405-685-2133, has a 2068 biorhythm program called "Life Cycles", and would like to hear from others with the same interests.

DAN PINKO C.6 Site 242, R.R.2, Parksville, Canada V0R 2S0, will repair your blown SDS load filter. This comes from turning off the power before unplugging the filter. The price is cheap.

BARY WINGERSKY PO. Box 185, Hopewell, NJ 08525, has rewritten the CAVE-MAN WP, for the 2068, and made several improvements. A good cheap word processor. He is also an excellent math statistician and available for consultation to help solve large statistical data problems.

Tom Woods, whose address is elsewhere in this issue, is always interested in talking to individuals interested in file management.

We had a list of survivors in last issue. Two magazines, which are not true survivors, but thrivers, are ZX COMPUTING, published by Argus

Anyone having trouble with the type size please get in touch with the editor (TB.) directly. We realize that some people have trouble with small text. Economics and content dictate this policy presently. We will probably do some experimenting with type size and headings in the near future.

BASIC TOPICS

IS IT A NUMBER?

Fred Nachbaur

The following routine contains a trap to insure that you enter a number. This is useful in math drill programs, to prevent cheating by entering an expression. The BASIC interpreter will accept the question as the answer by evaluating the expression for you. For example, in response to the question, "How much is 2+2?", by INPUTING 2+2, the computer will accept it as the correct answer. This routine eliminates the chance of the + being INPUT as part of a valid response. This also holds true for any character other than numbers.

Another trick in this program is the way it prints to the screen. It POKES the display file directly. To insure proper execution, this routine needs a padded out (full) display. (PEEK 16400+256*PEEK 16401)- (PEEK 16396+256*PEEK 16397) should equal 792. If you don't have 16K, then POKE 16389,100, to fake it. ☐

```

5 SLOW
10 LET B=PEEK 16396+256*PEEK 1
6397+780
20 LET E$="ILLEGAL ENTRY"
100 PRINT "ENTER A NUMBER"
110 INPUT D$
120 IF D$="" THEN GOTO 170
130 FOR C=1 TO LEN D$
140 IF D$(C)<>"." AND (CODE D$(
C)<28 OR CODE D$(C)>37) THEN GOT
O 170
150 NEXT C
160 GOTO 300
170 POKE 16418,0
180 FOR C=1 TO LEN E$
190 POKE B+C,CODE E$(C)
200 NEXT C
210 POKE 16418,2
220 GOTO 100
300 LET D=VAL D$
310 PRINT "D,";"THANK YOU"

```

Specialist Publications, 1 Golden Square, London, England W1R 3AB, and YOUR COMPUTER, Quadrant House, The Quadrant, Sutton, England SM2 5AS. ZXC is mostly games, but the current issue has a lot of good utilities. YC is a general use magazine, but the ZX material is generally excellent. They are great for 1000 owners. 2068ers will have to take some chances will machine code listings. Both of these magazines are available at some newsstands.

RADIO-ELECTRONICS has a 4 part hardware story in progress for doing something new with your Timex. If you haven't seen it, you'll need back issues to July. It is available at most newsstands. ☐

If you are an aspiring writer, and would like to review relevant software, send us a sample of your writing capabilities and interests. We'll follow up in due course. We have a lot to review.

2068 FELT HAT

Paul Bingham
Pleasantrees Programming
PO Box 7345
Mesa, Az. 85206

Paul sent along this little cutie. This is an excellent 3D play around program.

Z is the third dimension, running along the length of the hat, setting the lines apart by a certain amount. This value can be changed by the STEP value in line 30.

C is the value that gives the curvature to each Z value before it is plotted. A slight change in the .033 value in line 60, can have a dramatic effect.

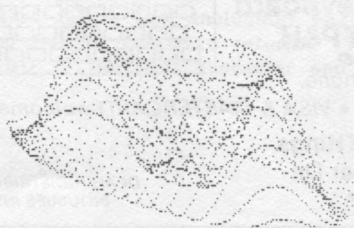
F holds the parameters for where each line starts and stops. Adjusting the initial and ending values of L in the loop in line 50, will dramatically adjust the size of the hat "brim". Altering the STEP value in line 50, will give you more or less points on each line. A higher number means a quicker run and therefore less time consuming for making adjustments.

Lines 70 and 80 hold the final values to be PLOTTed on the screen. Changing these values will center or stretch the hat in different ways. Have fun! ☐

```

10 REM      Felt Hat for 2068
20 BORDER 1: PAPER 1: INK 2
30 FOR Z=-142 TO 150 STEP 12
40 LET L=INT (SQR (ABS (1-(Z+Z
)/22500)))*175+.5)
50 FOR F=-L-3 TO L+4 STEP 4
60 LET C=SIN (.033*SQR (Z+Z+F*
F))*30
70 LET X=220-(240+Z-F)/2.5
80 LET Y=Z/5+110+C
90 PLOT X,Y
100 NEXT F
110 NEXT Z

```



MACHINE CODE TOPICS

2068 BASIC ROM CALLS

Ray Kingsley
Sinware
PO Box 8032
Santa Fe, N.M. 87504

This is the first of an occasional and indefinite series on machine-code programming for the 2068. I do not intend to take you through the rudiments of Z80 instructions, nor even the discovery and invention of the poked REM statement, but if you know how to do it even a little bit, then maybe I can give you a few ideas, or at least prep you for your next local Sinclair ROM trivia quiz.

It will help if you have some kind of machine programming aid at hand, a disassembler, an assembler, an efficient PEEK and POKE loop, or at least something more illuminating than a flashing K, in order to look at the code in the ROM. There are a few people that survived the 80's without HotZ, and it will not be a requirement to read this article.

I will be happy to receive your comments on my mistakes and omissions. I cannot promise you individual answers and snippets of code, but I will try to reply here to all items of general interest.

For unravelling a ROM, I prefer to start with the jump tables, which provide lists of the entry points of many of the main routines in the operating system. Once the main jump tables are understood, then it is generally possible to work out most of the auxiliary routines.

The way to find the main BASIC jump table in the ZX80's was to PEEK out the value of the syntax table pointer (T_ADDR), which Uncle Clive had thoughtfully labelled "very unlikely to be useful" in the ZX manual. I noticed that Timex removed that comment in the 2068 manual: have they succeeded in making it truly useless? Happily not. PRINT PEEK 23668 (and 23669) give 158 and 25, which convert to the hex address 199E. (Don't fiddle with multiplying out the decimals; just convert each byte directly to hex: 25d is 19H, etc.) Now 199E just happens to contain the address of the PRINT routine in BASIC, because I asked the machine to PRINT what I PEEKed. This address is near the beginning of the syntax table, which actually begins at 197BH. Finding the table this way saves hours of hunting.

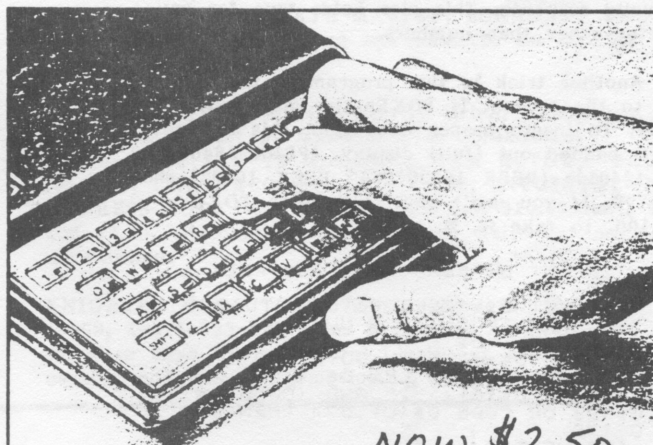
The syntax table is not a pure jump table. It contains a liberal scattering of "command class" bytes and indications of required separators for commands like the comma in PLOT X,Y. Nevertheless, it is not a difficult task to work out the valid addresses in the table and to start studying the routines at those addresses to see which BASIC commands they implement. (Well, knowing the ZX ROM does help a little.) Table 1 lists the addresses from this table for the 2068 BASIC commands.

I intend here to browse through the list to suggest some possible uses of these routines. However, using BASIC commands in MC is like trying metric tools on an American engine: some of them work, a little, for some things. Here are a few of the best; others can be worked out by analogy.

GOTO

On the ZX80 I wrote a BASIC disassembler and was amazed at its deliberative approach to the screen. To speed it up I started patching in machine-code routines. I needed routines that would take calls from anywhere and re-enter the BASIC program at a line to be decided in code. That was when I adopted the GOTO routine from Z80 code. In a "real" program the result is spaghetti, seeds of impossible logic, but if you are fond of potato racing and other such self-imposed handicaps, you might enjoy it.

The GOTO routine starts with a CALL to a very useful routine at 1F23, which loads the BASIC line number from the calculator (calc) stack into the BC register pair, rounding to the nearest integer. It then proceeds to put BC into HL. In the normal course of a machine code subroutine, you are not likely to have become deeply involved with the calc stack, so prepare first and enter this routine late with the initial value, the BASIC line number, in HL. That enables you to jump to the GOTO routine at 1EF6. The program at that point checks the validity of your line number (incorrectly) as less than 61439, and requires you to begin with the first statement. If you enter at 1EF8 with D already set to the statement number, then you can return to a particular statement (in D) of a line number (in HL).



TS 1000™/ZX81™ OWNERS:
"TOUCH TYPE" FOR JUST ~~\$5.95~~ **NOW \$2.50**

- Improve programming and game playing, too!
- Easy to install - no wiring or soldering required
- Clear Vinyl with key-shaped openings and adhesive backing.

TYPEX/81™ Keyboard

~~\$5.95~~ + ~~\$1.50~~ P&H

~~\$2.50~~ ~~\$1.00~~
Order by mail or phone.

216/234-2662 • VISA & MASTERCARD welcome.

GANHART/EARTHings

115 N. Rocky River Dr.
Berea, OH 44017

DEALER/DISTRIBUTOR
INQUIRIES INVITED

Table 1

BASIC COMMAND ADDRESSES IN THE 2068 ROM

GOTO	1EF1	READ	1D97
IF	1C5B	DATA	1E82
GOSUB	1F99	RESTORE	1E9D
STOP	1C59	DRAW	26DB
RETURN	1FD4	COPY	0A02
FOR	1C78	LPRINT	2155
NEXT	1D55	LLIST	1541
PRINT	2159	BEEP	0436
INPUT	222B	CIRCLE	2679
DIM	2FC0	OUT	1F04
REM	1B00	BORDER	243E
NEW	0D1D	DEF FN	201D
RUN	1F2B	OPEN	142A
LIST	1545	CLOSE	139F
POKE	1F0A	FORMAT	25CC
RANDOMIZE	1ED4	MOVE	25D0
CONTINUE	1EE4	ERASE	25D4
CLEAR	1F36	CAT	25C8
CLS	08A6	DELETE	20D1
PLOT	2635	RESET	2454
PAUSE	1FEB	SOUND	2128

Those below are not in the syntax table nor the floating point table:

LET	2EBD	INK	1BF9
STICK	2902	PAPER	"
FREE	2934	FLASH	"
SAVE	01AB (EXROM)	INVERSE	"
LOAD	"	OVER	"
VERIFY	"		
MERGE	"		

Ending your code for a USR call with:

```
LD D,02      Statement number
LD HL,26AC   9900 decimal
JP 1EF8      The GOTO
```

will put you back in BASIC starting with statement 2 of line 9900. If there is no statement 2, you will fall into the "Statement lost" error trap. The best use of this tactic would be to force a single re-entry point, independent of where the USR call was made, so as not to play havoc with program logic.

PRINT

You can print long strings to the screen by using the BASIC PRINT routine, and it will serve you fairly well for messages that you can set up at some fixed position in RAM. The rules are these: start with a quote (or comma/apostrophe); the address of that quote should be put into HL and passed to CH_ADD, which is BASIC's pointer for most commands. End with a quote and a carriage return (0D). Outside the quotes, the semicolon, apostrophe, and comma will work as they do in BASIC.

If you know how to format a floating-point form, or if you just write the PRINT statement in BASIC and then copy it up to your machine code area, you can use the AT, TAB, INK, PAPER and suchlike items as well. What you need in RAM are the sequence of characters that follow (not including) PRINT up to and including the 0D (carriage return). Then just point CH_ADD to the first character and jump to 2159:

```
LD HL, POINTER
LD (CH_ADD), HL
JP 2159
```

There is a more elementary approach to screen printing, the well-known Sinclair approach of loading the character code into the A register and doing an RST 10. All registers are preserved, and it's as fast as you're likely to need for most programs.

There are a couple of things you need to know to use the 2068's RST 10. First of all, you had better set up the right output channel, or strange lockups are likely. Channel 2 will give you the main screen, channel 3 the 2040 printer output, and channel 1 or 0 the lower (edit) screen. The way to set the channel is:

```
LD A, CHANNEL_NUMBER
CALL 1230      1,2,3
               CHANNEL
               OPEN
```

If you stay in machine code, the open channel stays open, but if you're jumping between BASIC and machine code, you should probably set it every time you start again in MC.

You will also want some kind of PRINT AT routine, so that you can RST 10 your character anywhere on the screen. The normal print-position set in the 2068 uses a "backwards" pair of line and column numbers, such that the top line is 24 decimal (18 hex) and the left column is 33 decimal (21 hex). It is often handy to use the following to invert a proper line and column position in BC into backward notation:

```
PRAT: LD BC, LINE_COLUMN (1,1 TOP LEFT)
      LD A,21
      SUB A,BC
      LD C,A
      LD A,18
      SUB A,BC
      LD B,A
      CALL 0914
      RET
```

The last call sets up the proper values in the system variables S_POSN and DF_CC. Use this as a set-up subroutine anytime you want a new print position. Do the LD BC, LINE_COLUMN in your main code and call the subroutine at PRAT. After the call, LD A,CHAR_CODE and RST 10 will put the character where you want it.

If you open the printer channel, LPRINT will work in a way similar to PRINT and RST 10 will output to the printer after you open channel 3.

INPUT

You can use the INPUT routine from MC in much the same way as you would use PRINT. Set CH_ADD to point to an area of RAM that holds the characters that follow an INPUT command, then jump to or CALL 222B (INPUT). The routine will prompt you just as BASIC does and will take your entry and put it into the proper place among the BASIC variables.

Your problem in machine code will be finding the input after the entry. Fortunately, there is a look-up routine in the ROM made just for the purpose, located at 2C70. Once again, CH_ADD is used as a pointer. You must have the variable name somewhere in RAM (e.g. A\$, X, UNKNOWN). Then point CH_ADD to the first character of the variable name and CALL 2C70. On return, for a simple string, HL will point three bytes short of the current value of the variable, and the two intervening bytes will hold the length. The following code will move the string to whatever address you set in DE.

```
LD HL, VARIABLE_NAME_POINTER
LD (CH_ADD), HL
CALL 2C70
INC HL
LD C, (HL)
INC HL
LD B, (HL)
INC HL
LD DE, WHERE_YOU_WANT_IT
LDIR
RET
LOOK FOR IT
GET LENGTH
MOVE IT
IF FINISHED
```

If the variable is numeric, you will have to deal with the floating point form (5 bytes). You can set HL to point to the first of the five bytes and CALL 3773 to move it to the calc stack. A numeric value follows directly after the address returned in HL after the lookup routine (2C70). Consult pages 256 and 257 of your 2068 manual for the format of other variable types. If you have a good single-stepper, step through the first three commands above and then look up the area pointed to by HL (as data, not code) to discover the offset to the actual variable value.

CLS

Here's one that's easy. Just call it for the traditional screen clear. However, if you are using the screen in machine code and have reduced the size of DF_SZ to 0 for full screen printing, then you will get a crash on the 2068. DF_SZ should be at least 2 for a normal CLS. You can increase it, do the CLS and set DF_SZ back to zero immediately if you choose.

PLOT

The PLOT command is one of those commands that gets its coordinates from the calc stack. It does so in the very first line of the routine with a CALL 2307, which loads the last value on the calc stack to B, the next last to C, and reports an error if either is over 255. To avoid fooling with the calc stack and details of floating point, all you have to do is to load BC with the coordinates of the point you want and to enter the routine at the second instruction at 2638.

The following little test routine should give you the idea by drawing a short diagonal line:

```
LD BC, 0133      STARTING COORDS
LD B, 0000        LENGTH COUNTER
LD C, 0000        SAVE ON STACK
DIAG: PUSH HL
      PUSH HL
      CALL 2638
      POP HL
      POP HL
      RET
      INC BC
      INC B
      INC C
      JR DIAG      PLOT AGAIN
```

DRAW

The DRAW command has two forms, with either two or three parameters. With just two parameters, DRAW will function without the calc stack, but with three parameters the calc stack is necessary, so it's time to learn how to put numbers there. The routine at 30E6 is STACK_A, and puts the number in A onto the calc stack. (For bigger numbers, not needed here, use STACK_BC at 30E9.) With both DRAW and CIRCLE, the interpreter gets the first two parameters onto the stack before the command routine begins. The first thing the command routines do is to look for the third parameter in a BASIC line, and again you will want to call in late.

The fifth line of the DRAW routine at 26E3 jumps for the two-parameter form to 27D0, which calls the actual line-draw routine and exits by the temporary-colors routine (0888). So the best way in will be at 27D0, but first it's necessary to set the current plot point by loading the system variable COORDS and to put an X and a Y displacement onto the calc stack.

For your TS2068

WINKY BOARD 2000

Cassette-Computer Interface

also for TS1000/1500, ZX81, ZX Spectrum

- Solve your LOADING problems
- Duplicate ANY TS/ZX cassette
- User friendly. Simply plugs into cassette player & computer jacks.
- Makes UPLOADER use easy

\$22.95 assembled/tested, shipping incl.

WINKY BOARD 2 for TS1000/1500, ZX80/81

\$18.95 assembled/tested; \$15. kit

SAS2000 SPEECH RECOGNITION SYSTEM

- Train your computer to obey your voice commands!
- Recognizes up to 8 spoken words
- Includes speech amplifier unit, cassette program, documentation

\$32.95 assembled/tested; \$27.95 kit

SRS1000 for TS1000/1500, ZX81. Specify

For your TS1000 Also TS1500, ZX81

WINKY BOARD 2 Cassette-Computer Interface

\$18.95 assembled/tested; \$15. kit

SAS1000 SPEECH RECOGNITION SYSTEM

\$32.95 assembled/tested; \$27.95 kit

COMPCOOLER Power supply-Computer

Interface solves most overheating problems \$6.95

H1 RES GRAPHICS PLANS plans to build interface \$5.

Utilities on cassette

KEY-unlock, merge, renumber \$ 9.

H1 RES PRINTER GRAPHICS \$9.

ZXL8 Fastload (not for 1500) \$10.

ALL PRICES INCLUDE SHIPPING

New lower prices are shown.



DREAMING OF RUNNING SPECTRUM SOFTWARE ON YOUR TS2068 ?

Now you can with

ROMSWITCH

Thousands of good British programs are available now from U.S. dealers

- Easy installation. No soldering, no drilling
- Plugs permanently inside your TS2068. Frees edge connector & cartidge port for other uses
- Simple magnet switch method selects TS2068 or Spectrum mode
- RUNS MORE PROGRAMS THAN THE "CHAMELEON"!

Price \$54.95 assembled/tested PPD

Catalog of all our products free on request.

G. RUSSELL ELECTRONICS

RD 1 • Box 539 • Centre Hall, PA 16828
814-364-1325 Mastercard/Visa 10am-8pm Check/MO

It works like this:

```
LD HL,0101      TO START LOWER LEFT
LD (COORDS),HL  SET START COORDINATES
LD A,80          X DISPLACEMENT
CALL 30E6        STACK IT
LD A,80          Y DISPLACEMENT
CALL 30E6        STACK IT
CALL 27D0        DRAW LINE
EXX              RESET HL'
LD HL,2B16
EXX
RET
```

The last four instructions are necessary if and when you return to BASIC. The DRAW routine uses the exchange registers, but the USR call also uses HL' to store the address 2B16, which must be there when your call comes back to BASIC. Try this reset whenever you've been making ROM calls from MC and get an Out of Memory or Nonsense in BASIC error when you come back to BASIC.

The three-parameter form of DRAW begins at 26E6 by reading the third parameter from the BASIC line to the calc stack. If you've already stacked the third parameter in your MC routine, then the proper entry point is at 26ED, where the floating point processor takes over.

A new problem is that the third parameter is the angle in radians. STACK_A will only stack integers, which doesn't give you much latitude in specifying the angle. Instead, you can take advantage of the routine at 3076, which will read the characters of a decimal number from RAM to the calc stack. The routine uses CH_ADD as a pointer, and it's a good idea to save and restore the original value of that pointer to prevent your BASIC program from getting lost. Put the characters of the decimal number somewhere in RAM and, for neatness sake, end it with an 0D (carriage return) so the end of the number will be clear. Then try the following:

```
NPTR: DB "3.58"      ANGLE IN RADIANs
DB 0D              END NUMBER
CURV: LD HL,3333      START POINT
LD (COORDS),HL     SET START COORDINATES
LD A,80            X DISPLACEMENT
CALL 30E6          STACK IT
LD A,80            Y DISPLACEMENT
CALL 30E6          STACK IT
RST 18             GET CH_ADD TO HL
PUSH HL            AND SAVE IT
LD HL,NPTR         NUMBER POINTER
LD (CH_ADD),HL     TO CH_ADD
LD A,(HL)          FIRST CHARACTER TO A
CALL 3076          STACK THE NUMBER
CALL 26ED          DRAW-CURVE
EXX
POP HL             GET OLD CH_ADD
LD (CH_ADD),HL     AND RESTORE IT

LD HL,2B16
EXX
POP HL
LD (CH_ADD),HL
RET
```

Be sure to call in at CURV, not at NPTR. Like BASIC, this routine will trap you if you run off the screen, so it might be best to get your parameters right with BASIC first.

CIRCLE

The CIRCLE command is very similar to the three parameter form of DRAW, but it does not require any initial setting of COORDS. The CIRCLE routine begins by checking for a comma and then reading in the third parameter (radius) from the BASIC line, so once again you should prepare first and call in late. The working part of the CIRCLE command begins at 2686, so try the following:

```
DB "3.58"      RADIUS
DB 0D          END NUMBER
LD A,80        X COORDINATE
CALL 30E6      STACK IT
LD A,80        Y COORDINATE
CALL 30E6      STACK IT
RST 18         GET CH_ADD TO HL
PUSH HL        AND SAVE IT
LD HL,NPTR     NUMBER POINTER
LD (CH_ADD),HL TO CH_ADD
LD A,(HL)      FIRST CHARACTER TO A
CALL 3076      STACK THE NUMBER
CALL 2686      DRAW THE CIRCLE
EXX            RESET HL'
LD HL,2B16
EXX
POP HL         GET OLD CH_ADD
LD (CH_ADD),HL AND RESTORE IT
RET
```

BEEP

I will leave an implementation of this command up to you. The routine at 0436 expects two numbers on the stack. You can put them on as integers with CALL 30E6 or by using the routine at 3076 to stack a decimal (complete with minus sign). Reset HL' before returning.

Another way of using the BEEP is to set DE and HL and CALL 03F3. DE should hold the frequency (Hz) of the note multiplied by the duration (sec), and HL is computed from the formula $(4,375,000/\text{freq}) - 30$, where freq is in Hz.

NEXT TIME

Not all of BASIC is listed in the syntax table. Most of the functions and a few commands are interpreted by the "floating point interpreter," a Sinclair specialty that is accessed by RST 28 in both the ZX and the 2068. RST 28 is just a jump to 371AH in the 2068. I browsed there for a few dozen bytes until I saw the LD DE,3696 at address 374D. 3696 is a valid and nearby ROM address; the "code" there disassembles as nonsense, looks like a jump table, quacks like a jump table. The functions there are listed in almost the same order as they were in the ZX. Learning to use these from machine code will give you some real programming power.

Next issue I'll try to explain some ways to use these functions and do painless floating point programming, with a couple of tricks that I haven't actually tried yet. If they don't work I'll think of an excuse and tell you about something else. ★



FORTH-WITH



By Gary Smith
Hawg Wild Software (C)1984
POB 7668, Little Rock, AR 72217

In Volume 1, No. 5 of SyncWare News I approached you with the idea of presenting an introductory/ tutorial column on the FORTH language. If you missed my introduction to FORTH article in Volume 1:3, I suggest you go back and read that. Contact SyncWare News for back issue availability if you need to. It will give you some good background information, and help explain why and how the pieces fit.

"Forth is like Pascal because they are both structured, with none of the 'can of worms' GOTOs and GOSUBs of BASIC and Fortran." True, except FORTH is structured bottom-up instead of top-down like Pascal. In FORTH you start with primitives (machine/ assembler- like basic words) and define increasingly complex words from them until the last word defined is your program.

"FORTH is like C because they both are designer-type languages." True, except FORTH is both extensible and interactive. In C, to use a new idiom you must link to it and recompile, whereas in FORTH you merely define the needed word. Further, FORTH is interactive. This means as you develop a new word you may test it in the interpreter mode just like you do in BASIC. You have no interpreted mode in C.

"FORTH is like LISP in that they are both used in artificial intelligence efforts." True, but this is really a LISP stronghold. It must be noted, though, that SAVVY from Excalibur in Albuquerque, NM is the only commercial AI system available for micros - at least as far as I know - and it is written in FORTH. Why? Because FORTH is compact. LISP is memory intensive!

By now you are asking, "Just what is FORTH like?" FORTH is like FORTH and nothing else - it's that simple. It has been shown that people who have never been exposed to a procedural language like BASIC, Fortran, Pascal, etc. grasp FORTH almost immediately. This is the one language, apparently, in which prior experience with other languages does not help - that experience may, in fact, hamper.

Since all SyncWare News readers are ace BASIC users we are resorting, next installment probably, to presenting FORTH in terms of BASIC. A lot will depend on your response. Want to learn FORTH? Come on in, have a seat. If you have a FORTH kernel, make your Sinclair/Timex a FORTH terminal and start playing with a very exciting language. Let me know what things you want me to bring to the party.

Thanks. Hope to hear from you soon. ✧



BASILS' COMPENDIUM

BASIC MACHINE CODE
FOR THE ZX-81/TS1000

John (Basil) Wentworth
1413 Elliston Drive
Bloomington, In. 47401

As the title implies, this series is designed to introduce the beginner to the basics of machine coding. The beginner in machine language that is. We must assume that you already have a working knowledge of BASIC.

As the title also implies, I will try as much as possible to relate machine code programming to the concepts of BASIC, with which you are already familiar. (I'm tempted to refer to this as Basil's Compendium on Machine Code, but I'll forego that conceit).

WHY MACHINE CODE?

Since you're already familiar with BASIC, you may wonder, why bother learning another, more foreign language? The answer of course is SPEED, and more economical use of memory, not to mention the fun of learning itself.

Think of computing as traveling in a foreign country. Machine language is the native language of the computer. If you have ever traveled abroad, then you are aware that is very convenient to have some knowledge of the local language. Although the residents of say, Barcelona (especially Barcelona!)

are very courteous, and will make every effort to understand your language, you'll get your point across a lot faster if you speak Spanish. Again, a big part of the fun of traveling is a chance to try the language.

It is the same with computers. By knowing BASIC (an intermediate language), you're already halfway there. BASIC is akin to Pidgin, which was developed to help English-speakers to communicate with Chinese and Melanesian people (that's your history lesson for today).

Let us use a system similar to that used by many modern language schools - we'll learn to 'say' something useful, without worrying for the moment about vocabularies and the rules of the grammar. Like the beginning language student, or someone playing the violin by ear, we'll know how to make that single statement, but we won't be able to apply that knowledge to any other situation for some time yet. Don't panic though, we'll learn the rules as we go along. It's just that you might not lose interest so fast if we teach you to do something first.

Start by keying in the rudimentary loader program shown in figure 1-1. SAVE this program. We're going to use it several times, and it's a better than even bet that you will damage it before you get through with this lesson.

THE 1 REM STATEMENT

The backbone of this machine code program is a 1 REM statement, which holds the individual instructions of the program. You won't recognize them unless you already know the language, but they're there just the same.

We start out with a rough draft of the statement; as shown in figure 1-2. Figure 1-2A shows the statement itself, while figure 1-2B tells you how it's built. The symbol '(GR S)' means 'Graphics S' (shift 9, shift S), and so on. With the statement roughed in, proofread it by GOTO 10. You should get the read out shown in figure 1-3. If not, then EDIT and make the necessary changes.

FILLING THE VACANCIES

There are some bytes of the program that can't be easily entered from the keyboard, which for the moment we have denoted by dots, to serve as placeholders. The subprogram at line 100 will help us to fill these vacancies with the proper numbers. Enter GOTO 100, and key in the numbers given in figure 1-4. When you have finished, input Z to break with an error 2/110. Finally, proofreading via GOTO 10, should get the readout of figure 1-5. Check it carefully, and correct any mistakes. (Use CONT to see the rest of the listing.)

You'll notice that a new column of information has been added. This column, which gives what are known as mnemonics, will not be on the display. It is given to help those of you who already know something about machine code to analyze the program. Don't worry if it all looks like gibberish to you. All will become clear in due course.

WHAT DOES IT DO?

So now that we have the program set up, what can we do with it? Well, try setting the cursor to line 150 and RAND USR 16514. LIST the program. Try it again with the cursor set on line 100. Add other lines, such as 2000 REM, 6900 REM, etc., and try lopping them off, one by one.

I call this program PROGTOP, because it sets a top on the program (similar to ramtop for the top of ram), and it can be very handy. So now you have an idea of the economy and speed of machine code. Remember, the program for all of that lopping off is in the 27 bytes of the 1 REM. Bit by bit, as you read this series, you'll find out what's going on in that line.

There's one big difference between your computer and a resident of Barcelona. If you say 'soy listo' in a restaurant on the Ramblas, the waiter will smile and say to himself: "This turista says that he is a clever guy, but que va! He really means 'estoy listo' - that he's ready." It's NOT the same with your computer. As in BASIC, since the computer is a literal minded creature, it does what you tell it and not what you meant. What you say is what you get. There's a fortune waiting for the person who invents a workable DWIM (do what I mean) button. The trouble is, just as machine code is more powerful than BASIC, the significance of your mistakes is multiplied as well.

EQUIPMENT NEEDED

The material in this series is designed for all ZX-80(8K) through TS1500 computers (some of the listings will work on the 2068 with appropriate address changes although untested at this time). Please read ZX81 to include the entire family.

Since machine code deals directly with the CPU - Central Processing unit - the basic principles here apply to any Z80 based computer. However, access to the CPU may be different from one computer to another. You should have 16K or more of RAM in order to use the loader programs to be described, along with tape or disc equipment for SAVEing programs. I strongly urge you to use a notebook and pen to keep permanent notes. If you learn anything at all from this series, then you're likely to find yourself consulting your notes again and again.

STUDY MATERIALS

There are two books that I have found invaluable. One is the instruction manual that came with my ZX81, especially the few pages that list the ZX81 character set (ZX81ers beware of the graphic character typo in your manual). The other is the outstanding "Programming the Z80" by Rodney Zaks (Sybex Inc., 2344 Sixth St. Berkely, Ca. 94710). This is a scary book, because of it's large scope. I hope however, that by the time you have finished reading this series, you'll find Mr. Zaks' volume comprehensible as well as comprehensive.

One other study technique is to get out all those magazine articles on machine code programming that you passed over the first time around. Read them again, in the light of what you will be learning here. Analyze the programs presented, adapt them to your own needs and improve them if you can. See you next time. 𐀀

```

10 REM PROOFREADING THE PROGRA
M
20 FOR F=16514 TO 16527
30 PRINT F;TAB 7;PEEK F;TAB 11
;CHR$ PEEK F;
40 PRINT TAB 16;(F+14);TAB 23;
PEEK (F+14);TAB 27;CHR$ PEEK (F+
14)
50 NEXT F
60 STOP
100 REM POKEING THE VACANCIES
110 INPUT A
120 PRINT A;TAB 6;
130 LET A=A+16500
140 INPUT B
150 POKE A,B
160 PRINT PEEK A
170 GOTO 110

```

FIG. 1-1. The Rudimentary Loader

```

1 REM ...ANDY.WS.RND7.4.7. FA
ST 7...PI LPRINT S..TAN

```

A. How It Looks

```

1 REM . (GR S) AND Y . W S
(GR W) AND 7 4 . 7 . FAST 7 .
. . PI LPRINT S . . TAN

```

B. How It Is Done

FIG. 1-2. 1 REM for Progtop - First Draft

... COMPENDIUM

```

16514 27 . 16528 35 7
16515 27 . 16529 27 .
16516 10 . 16530 229 FAST
16517 64 RND 16531 35 7
16518 62 Y 16532 27 .
16519 27 . 16533 27 .
16520 60 U 16534 27 .
16521 33 S 16535 66 PI
16522 130 L 16536 225 LPRINT

```

```

14 237
15 75
19 117
27 252
29 86
32 94
33 235
34 237
38 241
39 119

```

FIG. 1-4
Poking The Vacancies

```

16514 237 GOSUB LD BC,
16515 75 ? (16394)
16516 10 .
16517 64 RND
16518 62 Y LD A,
16519 117 ? 117
16520 60 U INC A
16521 33 S LD HL,
16522 130 L 16514
16523 64 RND
16524 35 7 INC HL
16525 190 . CP (HL)
16526 32 4 JR NZ,
16527 252 UNPLOT -4
16528 35 7 INC HL
16529 86 ? LD D, (HL)
16530 229 FAST PUSH HL
16531 35 7 INC HL
16532 94 ? LD E, (HL)
16533 235 FOR EX DE, HL
16534 237 GOSUB SBC HL, BC
16535 66 PI
16536 225 LPRINT POP HL
16537 56 S JR C,
16538 241 LET -15
16539 119 ? LD (HL), A
16540 201 TAN RET

```

FIG. 1-5. Proofreading the Final 1 REM

FIG. 1-3. Proofreading the Provisional 1 REM

2050 MODEM with Smart I Software

Designed especially for all TIMEX computers, the 2050 offers some exceptional features at a remarkable price. It's one of the new "modular" hook-up modems. This means you don't need to put acoustic "cups" on your telephone. Just plug the modem into your phone jack. The standard SMART I software that comes on cassette will allow you to use the 2050 with TIMEX 1000, 1500, and 2068 computers and provides the capability to auto answer incoming calls as well as access any outside data base.

SMART II Software for 2068

The optional SMART II software comes on cassette and provides auto-dial of up to 14 different phone numbers, each with its own preset access codes and log-on sequences! You can save and load other files containing phone numbers and other communication parameters...making the auto dial feature capable of many more than the basic 14 numbers. It will also upload and download to and from memory.

TASMAN PRINTER INTERFACE

A Centronic standard parallel interface which includes a 3-foot cable, "LLIST and "LPRINT" software. Also includes FAST M/C, HI-RES screen copy routines, available for the Epson, Star, Seikosha (Gorilla Banana) and Radio Shack color printers. "IN COLOR"!!

TASWORD TWO WORD PROCESSOR

The professional standard word processor for the 2068 computer system. A superb word processing program which is equal to Wordstar™ (some say better). Interfacing is a cinch since Tasword Two drives full-width printers via the Tasman interface.

TASWIDE PRINT UTILITY

Allows your "BASIC" programs to output the display in 64 columns to your TV or printer!

UPLOAD 2000

TS1000 to TS2068 Program Converter...This new program allows you to convert your library of ZX81, TS1000 and TS1500 programs into TS2068 programs! Now you don't have to rekey all those programs to make them compatible with TS2068s. It works for programs written in BASIC, not machine code. Due to the differences in recording methods used, we recommend the use of a filter (FL1000) when loading TS1000 programs into your TS2068.

FL1000

5 1/4-inch DUAL FLOPPY DISK DRIVES

CLOSE OUT SALE - brand new single-sided, double-density Dual Floppy Disk Drives that includes power supply on/off switch, case and fan. (NOTE: You must supply the required controller board and cables to complete the system.)

DOS CONTROLLER BOARD & CABLE for TS1000/1500

The above drives can SAVE & LOAD to disk or cassette type from T/S BASIC commands using this COMPUSA Controller board. Data files can be created using READ & WRITE physical track commands. Extremely fast-loads a 32K program in 1 second. Extremely reliable - all files read & checked after being written.

PROGRAMMERS REFERENCE CARD for ZX80, ZX81 & TS1000

Includes: memory maps, complete summary of Z80 language and more.

MANY MORE PRODUCTS AVAILABLE

To get information send self-addressed stamped envelope.

E-Z KEY

SUITE 75TW

711 SOUTHERN ARTERY

QUINCY, MA 02169

TELEPHONE (617) 773-1187



GAMES TO LEARN BY, INC.

David Dubay
P.O. Box 78
28 Claire Hill Rd.
Collinsville, Ct.
06022
203-673-7089

Mail Order
Write or Call

Charles Warner
P.O. Box 575
2 South Street
Williamsburg, Mass.
01096
413-268-7505

Software or Hardware

3D ** DEATHCHASE \$19.95 **



1983 Games to Learn By,



T/S 2068 Computer \$120

VU-3D 16.00

VU-FILE 16.00

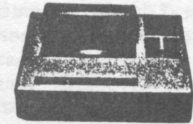
CRAZYBUGS 11.00

* FROGGER by Sega 19.95*

FLIGHT SIMULATOR 16.00



T/S 1500 COMPUTER \$75



T/S 2040 PRINTER

\$75.00

FOR THE SURVIVORS FLYER \$1.00
OR TO ORDER:

Mass. & Conn. residents Please include tax

CHECK ONE: ☐ Payment enclosed. ☐ MasterCard ☐ Visa

Card No. _____ Exp. Date _____

Mr. _____
Mrs. _____
Ms. _____
(please print full name)

Address _____ Apt. _____

City _____ State _____ Zip _____

DEATHCHASE



DEATHCHASE by M J Estcourt
The Story So Far

It is 2501, 100 years after the Great War. The North American continent is ruled by MIGHTY warlords in constant conflict over forest territory. You are one of the elite mercenaries, Riders of the BIG BIKES. It's a quick way to get rich--and a quicker way to die. You patrol your forest day and night, chasing enemy RIDERS and destroying them with your guided photon bolts for \$1000 a time. You may find helicopters and tanks too--your masters reward you particularly well if you destroy these.

VU-3D	Cass 16.00
VU-FILE	Cass 16.00
Wordcross	Cass 11.00
Word Play II	Cass 11.00
Personal Portfolio	Cass 11.00
Horace Goes Skiing	Cass 11.00
Horace & Spiders	Cass 11.00
Language Usage	Cass 11.00
Penetrator	Cass 11.00
Fun Golf	Cass 11.00
States & Capitals	Cart 17.00
Budgeter	Cart 18.00
Flight Simulator	Cart 20.00
Androids	Cart 18.00
Crazybugs	Cart 16.00
Casino I	Cart 12.00



Art for All Ages. Another good program for the beginning designer is Art for All Ages for the Timex 2068. This software lets you start drawing with circles and squares. Later, you can expand to create pictures in free-hand using a joystick. Art for All Ages features eight colors and a "whoops" key that lets you erase a line you've just created. You can even alter a picture after you've saved it. There's also a command that lets you pluck out part of the picture and move it to somewhere else on the screen, or remove it altogether. Art for All Ages is currently available from Games to Learn By at PO Box 575, Williamsburg, MA, 01096, and costs \$15.95.

***** GTLB'S *****
* NEWS RELEASES for TS/2068 *

We have the manufacturing rights for PSION'S Vu-File (\$16.00), Vu-3D (\$16.00), Vu-Calc (\$16.00), Flight Simulator (\$16.00), also Sega & Cornsoft's FROGGER (\$19.95).

T/S 2068
EXCLUSIVE RELEASE
\$19.95



CHESS	Cass 11.00
VU-CALC	Cass 16.00
Word Play I	Cass 11.00
Capitalization	Cass 11.00
Math Wizardry I	Cass 11.00
Math Wizardry II	Cass 11.00
Home Improvement	Cass 11.00
Hungry Horace	Cass 11.00
Quadra-Chart	Cass 11.00
Crossfire	Cass 11.00
States & Capitals	Cass 11.00
Budgeter	Cass 16.00
Flight Simulator	Cass 16.00
Androids	Cass 11.00
Crazybugs	Cass 11.00

TRANSLATIONS

By Fred Nachbaur

Memotext to ZX Pro/File

Required hardware:

ZX81/TS1000/TS1500
Memotext W/P module
64K RAM or 32K + Hunter board

Required software:

Hot Z or Hot Z II, 16K version
ZX Pro/File, 16K version

If you have a Memotext word-processor module and 64K RAM, you undoubtedly have already made use of its powerful "text" files mode. Memotext also supports "data" files, organized in six fields per file, up to 31 characters per field. Entering data files is just as easy and fast as text, thanks to the re-defined keyboard, visually logical upper/lower case screen, and cursor editing. Unfortunately, manipulation of data files is very limited; M/text is simply not meant as a data-base program. Wouldn't it be nice if you could enter your data files using Memotext, and then transfer them into your favorite filing program for sorting, searching, etc.?

I ran into the need for this when sending our subscriber list (on Memotext data files) to Tom Woods, who did not have a Memotext. So what more logical a program to try this out on than his ZX Pro/File? Fortunately, this is quite possible, and not as difficult as it might seem at first. The biggest problem is loading M/text files into your computer without M/text on-line. It may be possible to patch the Memotext load routine into RAM to do this directly, but meanwhile it's "the universal solvent", HotZ, to the rescue. We can use HotZ to load the Memotext data file into high memory. On quitting HotZ, load ZX Pro/File, and then translate and transfer the files into its' data string (D\$.)

Before we get on to the actual mechanics, let's look at how the two programs differ in their file structure. Memotext uses CHR\$ 255 (FFh) to mark the beginning and end of the files, and CHR\$ 254 (FE) as end-of-field markers. There are no end-of-file markers after each file, since each file is fixed at six fields; every sixth FE marker is interpreted as the start of a new file. Empty fields result in the FE marker (only), much in the style of the ZX display file when "collapsed" and containing empty screen lines. ZX Pro/File, on the other hand, is capable of handling any number of fields per file, so an "end-of-file" marker is necessary (CHR\$ 23) for each file. As a result, there is no need for end-of-field markers for empty fields. Also, ZX Pro/File uses CHR\$ 140 for end-of-field instead of CHR\$ 254. So to translate to ZX Pro/File, we have to:

- 1: Change end-of-field markers from CHR\$ 254 to CHR\$ 140
- 2: Change every sixth end-of-field CHR\$ 140 to an end-of-file CHR\$ 23.
- 3: Change the marker at the end of the entire set of files from CHR\$ 255 to CHR\$ 134.
- 4: Since Memotext reverses the cases (normal = capitals, inverse = lower case) compared to the printer interface, our translation routine also has to change inverse to normal and vice versa.
- 5: Any apostrophes (inverse comma or CHR\$ 154) have

to be changed to inverse colon (CHR\$ 142) due to a quirk in the CIF/Memotext system. (Otherwise, when printed, the apostrophes are reversed.)

If your existing M/text files contain inverse pound signs (prints as "%") or asterisks (*) then you'll have to additionally change them to something else, or they will be falsely interpreted by ZX Pro/File as end markers. For the sake of simplicity, and since it is rare that these would be used for typical filing jobs like mailing lists, etc. this feature is not included in the program presented here.

So with no further ado, here's how you go about getting data files into ZX Pro/File via Memotext.

- 1) Power up your 64K computer, then POKE 16389, 107 then NEW. This sets RAMTOP lower and insures that your file length will not exceed the 11000 character maximum dictated by the 16K ZX Pro/File. Type in your data files, then save to tape using "SDF." Rewind the tape.
- 2) Turn off Memotext, and reset the computer with your reset switch if you have one, or by powering down and up again. This time, leave RAMTOP at its default value of 16K (do not POKE 16388 or 16389 first). Turn on your 8-16K region, this will be where the translation and transfer routines will be stored.
- 3) Load the 16K version of Hot Z or Hot Z II (I used HZII) and use it to enter the machine code translation routines as listed. I wrote it to take the area from 2000h (8192) to 2086. You may wish to locate it elsewhere in the 2000-3FFF region. You will have to change the CALLs and variables references. When entered, use HotZ to save this code to tape for future use. (If you're entering the program as listed, put END at 2087, cursor at 2000, and SAVE.)
- 4) Use Function 0 to clear the area from 8000-BFFFh. Now put your M/text tape back into the recorder, set cursor to 8020h and END to 8021, then LOAD. When the first brief load pattern (Memotext header) has loaded, the screen will reappear; stop the tape in time so it is still positioned before the main body of the data. Locations 8020 and 8021 now contain the length of the data file, in low-high order. Set END to 8020 + this value; e.g. if 8020 contains 3B and 8021 contains 29, set END to (8020+293B) = A95B. With cursor at 8020, give the LOAD command and restart the tape.
- 5) When the tape has loaded, the screen will reappear and you can look through the data file. The first byte (at 8020) identifies the type of file (data or text) and should contain a 29h ("D"). The next two contain length-of-files minus 2. The next 60 bytes are for the field labels. After that is an FFh to mark the start of file. Take a look at the location just before END to verify that it contains the required FFh to mark end of files.
- 6) You may now quit HotZ. If you have 64K memory, you can then RAND USR 8192 to store HotZ (or any 16K program, for that matter) in the 48-64K area (C000-FFFFh) complete with system variables, display file, etc. Later, you can instantly recall HotZ or other software using RAND USR 8204 (with HotZ, first POKE 16388,128 and POKE 16389,127 then NEW before calling

USR 8204, since it sets RAMTOP 128 bytes lower to protect its variables. Also call USR 8204 from the same mode, FAST or SLOW, as you used when you called USR 8192.)

7) LOAD the 16K version of ZX Pro/File. When loaded, STOP operation, and enter LET P = USR 8233. This routine will translate the files in less than a second in FAST mode, and will return the correct length-of-file into BASIC variable P (ZX Pro/File's length variable = actual file length plus 21.)

8) At this point, the files have been translated, but not yet transferred into D\$. RAND USR 8216 to do the actual transfer. Now you may GOTO 17 to restart ZX Pro/File, and you can use any of the program's options to process the files any way you want. Use the SAVE option to record the program and data on a separate tape.

Note that our routine leaves end-of-field markers even after blank lines; while this is somewhat non-standard for ZX Pro/File, it doesn't seem to mind if they're there. So far, I haven't run into any difficulties. The programming would have been more longer and more complicated if we had to get rid of them, and the only advantage would have been a slight space savings - not enough to justify the additional trouble.

Next time, I'll try to have a similar procedure worked out to translate M/text data files into the popular Psion Vu-File program. In subsequent installments we'll work out programs to go the other way, from Pro/File or Vu-File to Memotext, so you can write form letters, etc. around existing files and make the interaction between your WP and data base as complete as possible. Meanwhile, happy "Memo/Filing!" ☺

```

=====
ADDR  HEXCODE  NAME  MNEMONIC
=====
2000  010040  UPLD  LD BC,ERNR
2003  1100C0      LD DE,C000
2006  210040      LD HL,ERNR
2009  EDB0      LDIR
200B  C9      RET
200C  010040  DNLD  LD BC,ERNR
200F  110040      LD DE,ERNR
2012  2100C0      LD HL,C000
2015  EDB0      LDIR
2017  C9      RET
2018  010000  XFER  LD BC,0000
201B  2A1040      LD HL,(VAR5)
201E  111B00      LD DE,001B
2021  19      ADD HL,DE
2022  EB      EX DE,HL
2023  215380      LD HL,8063
2026  EDB0      LDIR
2028  C9      RET
2029  210080  XLAT  LD HL,8000
202C  7E      LD A,(HL)
202D  23      INC HL
202E  FEFF      CP FF
2030  2802      JR Z,2034
2032  18F8      JR LP01
2034  222420      LD (2024),HL
2037  E5      PUSH HL
2038  0E00      LD C,00
203A  7E      LD A,(HL)
203B  FEFF      CP FF
203D  2807      JR Z,FINI
203F  CD5820      CALL EXAM
2042  77      LD (HL),A
2043  23      INC HL
2044  18F4      JR LP02
2046  3E86      LD A,86
2048  77      LD (HL),A
2049  23      INC HL
204A  D1      POP DE
204B  AF      XOR A
204C  ED52      SBC HL,DE
204E  221920      LD (2019),HL
2051  111300      LD DE,0013
2054  19      ADD HL,DE
2055  E5      PUSH HL
2056  C1      POP BC
2057  C9      RET
2058  00      EXAM  NOP
2059  FEFE      EFLD  CP FE
205B  200F      JR NZ,AP0S
205D  3E8C      LD A,8C
205F  0C      INC C
2060  57      LD D,A
2061  79      LD A,C
2062  FE05      CP 05
2064  2004      JR NZ,206A
2066  0E00      LD C,00
2068  1617      LD D,17
206A  7A      LD A,D
206B  C9      RET
206C  FE9A      AP0S  CP 9A
206E  2003      JR NZ,IURS
2070  3E8E      LD A,8E
2072  C9      RET
2073  FE86      IURS  CP 86
2075  3807      JR C,NRML
2077  FEC0      CP C0
2079  3003      JR NC,NRML
207B  C680      ADD A,80
207D  C9      RET
207E  FE26      NRML  CP 26
2080  D3      RET C
2081  FE40      CP 40
2083  D0      RET NC
2084  C680      ADD A,80
2086  C9      RET

```

: Store program
at 49152-65535d

: Get stored program
into 16-32K area

```

: # bytes to move in BC
: get VARS start address
: and add 27
: for start of file.
: destination in DE
: source address in HL
: transfer
: return
: Init. address = 32768d
: get char. at add. into A
: next address
: is it start of file?
: yes - go on
: no - loop back
: put source add. into transfer
: routine
: init. field counter
: get char. at address into A
: end of files?
: yes - go to end routine
: no - translate char.
: put translation into add.
: next address
: loop back
: put end-of-files mkr. (134d)
: into address
: next address
: get start add. into DE
: reset carry
: calculate # of bytes
: and load into transfer routine
: add 19d
: to # of bytes
: for Pro/File variable P
: and put into BC
: and return
: end-of-field?
: no - go on.
: yes - make it 140
: increment field counter
: store character
: field counter in A
: is it 6?
: no - go on
: yes - reset counter,
: change character to 23d
: and put back into A
: return
: is it apostrophe (inv. comma)?
: no - go on
: yes - make it inv. colon
: return
: is it inv. (upper case) letter?
: no - go on
: no - go on
: make it "normal."
: return
: is it normal (lower case) letter?
: no - return
: no - return
: yes - make it inverse
: return

```

A VIDEO UPGRADE ?

Why? Is the predominant first question. If you find that your computer does its' job, but you would like to see it work a lot faster (in SLOW), if you tired of that annoying fast mode flicker, if you have a program longer than 16K of BASIC and want it to run without crashing, then this is for you. Have you put mucho time and money into your little computer? Do you not want to spend a lot more to get a bigger computer and start from scratch again? Please keep reading.

Among other advantages, this project gives you the following enhancements:

1. No FAST or SLOW screen flicker.
2. SLOW runs 5.5 times faster (almost as fast as FAST).
3. No display file crashes at all. Memory restrictions are removed for BASIC and machine code (with opt. mod to some memories).
4. COLOR on the screen and LOWER CASE letters on the screen and 2040 printer. This system can be made compatible with the Memotech I/F additional decoding required and others.
5. The video output is 75 ohm composite video.
6. No additional memory space is required (I/O

mapped), and return to standard Sinclair video is as easy as unplugging the boards (some mods may change this).

7. The IX register is available for programming and memories using the Z80 refresh register will run 'in spec'.

8. Almost all software will run as is, although some MC programs will run too fast (I love it!). Basic games become exciting to play.

9. This project is 8K ZX80 compatible if it is not previously upgraded.

10. The advanced capabilities of the TMS9918 are at your disposal if you want a challenge in machine code.

Remember, this is the same chip that the Adam, TI99, MTX512 (Memotech) and the new Japanese MSX type use, for video display.

This project is a major undertaking and will be divided up into two installments plus additional information as we uncover it later. For those of you who are board builders, we have included exact size board layouts and finished pictures. JLOs' prices are excellent however, for those of you who want to 'kit' it (finished photos in next issue).

TMS9918A VIDEO UPGRADE PART 1...

JOHN L. OLIGER
11601 Whidbey Dr.
Cumberland, In. 46229

The TMS9918A video project consists of two pc boards, a small modification/addition to the main computer board, and a +5, +12, -5 VDC power supply (an expansion board of some type is necessary). Video board "A" contains the VDP chip, 16K video ram and one support chip for I/O interfacing. Board "B" contains the TMS9918 enhanced BASIC 2764 eprom, a switchable 2764 eprom (8-16K) socket and the logic necessary to decode the system and bank switch the character generator space (for initialization and printing to the 2040 printer).

The computer board modification consists of a diode OR-ing the NMI input to the Z80, allowing the VDP to interrupt the CPU instead of the ULA chip. This is a transparent modification and does not hinder the normal video display when the boards are removed.

The power supply is necessary for the video ram. Although power requirements are modest for -5 and +12 VDC and heat sinking is not required, the +5VDC regulator should have a hefty heat sink especially when using a TS 16K rampack. When using any ram (TS1016, memotech 64K) or other boards that require 9 VDC on edge trace 2B (see edge connector drawing), the following 'mod' will allow this supply to power the whole computer:

On the computer end of your expansion board, cut the trace going to the computers' edge trace 2B. Connect four (4) 1N4148 diodes in series (anode to cathode), then connect the anode end to the lower left trace (looking at it from the computer end) on the expansion board (12 VDC), and connect the cathode end to trace 2B on the expansion side of the cut.

NOW, LET THE FUN BEGIN!

First, install the diode/resistor on the computers main board. a 1N4148 diode and a 4.7K ohm resistor (yellow/violet/red). Check the main computer drawing for your type of computer and board. Make the necessary cuts and connections.

Reassemble your computer and test it to see if it still works. If it does not, then go back over it and check for solder bridges, etc., in the area surrounding your artwork. Do NOT proceed until your computer works with this 'mod' installed (ZX80 with a ROM CS not mod installed).

Next, have your power supply ready for hook up. -5 VDC is connected to the top left trace of the expansion board (22AWG wire or larger). This is just left of the TS trace 1A (remember, Johns' expansion board is a true 50 pin board). Connect the 12 VDC line to the bottom left 'extra' trace and the +5VDC line to the 5 VDC Sinclair trace (1B). Connect the power supply ground to the expansion ground (traces 4B and 5B). Hook up the 4 diode connection as previously stated, and you are ready for a cool running, externally powered computer. You won't have

to use that 9 volt plug on the side of your computer anymore. Check all connections and power up your computer. If it does NOT work, then power down immediately and check your connections again. Look for the proper edge connections and shorts to adjacent tracks. Check for proper voltage on each connected line.

You have now made the necessary computer board mod and power supply hookup. Before we begin the video boards, take a moment to relax, get an old issue of BYTE or RADIO ELECTRONICS and look up a source for the TMS9918A chip and its' companion the 10.738635 MHz crystal.

The following are sources that we know of:

JDR Microdevices 1-800-538-5000
in Calif. 1-800-622-6279
ACTIVE Electronics (Mass.) 1-800-343-0874
MILGRAY Electronics (with a company PO) is in
Wash. DC. and has the VDP chip for \$25.00,
purchase order only.

JDR and Active has it for \$39.95 and \$32.95
respectively, but Active is VERY, VERY SLOW!! The
crystal is about \$4.00

VIDEO BOARD "A" ASSEMBLY INSTRUCTIONS

Assuming that you make your own boards, the full size layout is available for board "A" and "B". It is beyond the scope of our text in this article to go through the steps required to make PC boards. We can supply those of you who are interested with information on PC assembly, if you supply us with a SASE.

Assuming at this point, that you now have possession of board A, These instructions will get you through its' assembly.

1. Install all feedthroughs on the board using 30 AWG solid wire, in every location that has a round "doughnut" pad, on the COMPONENT side of the board. DO NOT install feedthroughs in the rectangular pads on U1s' socket, pins 21-24.

To install a feedthrough, insert the wire from the boards' component side, to extend about 1/8 inch outfrom the non-component side of the board. Bend the wire over on both sides, lay the board down flat (non component side up) and solder the wire on this side. Turn the board over, cut the wire off about 1/8 inch above the board, bend the wire over and solder this side also. Repeat this until all the feedthroughs are in place. Clean the flux from the board with some acetone or flux remover and inspect your work for bad solder joints or shorts. Touch up as necessary.

2. Install resistors R1, R2 and R3. Next solder all IC sockets onto the board.

USE CARE in soldering U3-U10 (16K RAM) sockets. The memory array is VERY prone to shorts. Clean the flux off and again check your work.

3. Install transistor Q1 by bending its' center pin forward (toward flat front) and placing it in the EBC pads with the flat facing the boards' left. Solder in place.

4. Install C1, C2 just left of R1, in their capacitor marked holes. Install C3 just right of Q1, denoted by the standard polarized capacitor symbol. Be sure the positive pin of C3 goes in the hole marked with the "+" symbol. If C3 is crowded by R2, then it can be laid down with its' top toward the edge connector. Bend the leads as necessary and solder in place.

5. Install the 10.7386 MHz. crystal by first cutting the leads to 1/2 inch length. Insert the leads only far enough through the board, at its' location marked with a crystal, to solder them in place. Twist the crystal 90 degrees and tape it down (gently) to the board, shaping its' leads as necessary. Clean and check your work.

6. If it all looks good, then flip the board over and install the 7 tantalum bypass caps (Cbp) on the non component side (carefully).

Cbp	PLUS END	MINUS END
1.	+ U1-pin 33	- U1-pin 12
2.	+ U2-pin 14	- U2-pin 7
3.	+ U10-pin 16	- U8-pin 1
4.	+ U3-pin 16	- U9-pin 1
5.	+ U6-pin 9	- U6-pin 16
6.	+ U8-pin 8	- U8-pin 16
7.	+ U3-pin 8	- U4-pin 16

Bend each caps leads to fit over their respective pads, cut lead off 1/8 in. past bend, tin each end and tack down one end to hold it in place. Solder opposite end followed by the tacked end. Repeat for each cap. Inspect, trim and touch up your work.

7. Install jumper wire from U1-pin 33 to U6-pin 9.

8. Prepare a small length of coax cable by stripping each end so that 1/4 in. of the center conductor extends from the center insulation and 3/8 in. of wire braid, twisted together, extends 90 degrees out from the cable. Solder the center conductor of one end to the "VIDEO OUT" hole, and the braid to the ground hole on the right. Tin the braid and center conductor of the other end and connect appropriately to a female RCA (phono) jack. Clamp the cable to the board by forming a 'U' from a piece of bare 22 AWG wire, slipping it over the cable and through the two holes on the right side. Bend the wire over and solder, Clip off any excess lead.

9. Jumper U5-PIN 9 to Vcc(+5VDC). Pick up Vcc just above and to the right of the edge connector, at the second edge trace from the right, on the non-component side of the board.

10. Install all ICs on the board. The VDP has pin one to the left and the rest have pin one DOWN. TAKE A BREAK!



Be sure to check the board before going any further. See section on testing the boards.

VIDEO BOARD "B" ASSEMBLY INSTRUCTIONS

1. As with board 'A', it is necessary to solder all feedthroughs on the board. Use 30AWG wire in every round 'doughnut' pad on the component side. Do not install a feedthrough in the square pad, just left of Eb. Clean the flux from the board with acetone and carefully inspect your work for shorts to adjacent traces and bad solder joints. Touch up anything you see that looks at all suspicious.
2. Install resistor R1 in its' holes, where marked on the component side, just left of Eb. Be sure to solder its' right lead on both the top and bottom of the board, at the square pad. Since R1s' holes are so far apart, take care to shape the resistors' leads to fit and solder in place.
3. Install all IC sockets onto the board. Again, as with board 'A', take care in soldering the two 28 pin sockets in place. Clearances are tight. Clean the flux from the board, inspect and touch up your work once again.
4. Install SW1 in its' mounting hole, just left of Ea. These holes are marked on the boards component side with an 'x' between them. Install the switch so that it is closed (on) when it is up, and open (off) when it's down. It should work like a normal room light switch.
5. Install the two tantalum type bypass capacitors in their locations on the boards' non-component side, in the locations detailed below. Install the capacitors in the same manner explained in step 6 of the board 'A' assembly instructions. Remove all flux with acetone and Q-tips, and check your work once again for shorts and bad solder joints.

CBP1 + U4-pin 14	- U4-pin 7
CBP2 + Eb-pin 28	- Ea-pin 14
6. Insert all ICs into their sockets. All four logic chips mount with pin 1 DOWN, and the eeprom(s) mount with PIN 1 TO THE RIGHT. Eb is optional and not needed for the project to work. It may be left empty. Ea will contain the TMS9918A Video enhanced BASIC for the project.



Be sure to check the board before going any further. See section on testing the boards.

The machine code necessary for this project will be covered in depth in the next issue. If you have no facilities for burning eeproms and/or just don't want the hassle, we can supply you with an eeprom with the previously noted (SWN 1/4) improvements and enhanced BASIC for \$20.00. Please specify your printer interface, Timex, Memotech, or JLO (only one per eeprom). This project is compatible with Memotech I/F only with additional decoding on the video board, and an eeprom with the I/F code patched in. If you are in need, please get in touch with me in the evening. Tom Bent.

John supplies a tape for \$6.95 that contains all the necessary MC for those of you who have an eeprom programmer and 21 VDC power supply.

Those of you who still have the OLD 8K rom that was updated two years ago, will have to update before the machine code will work properly. Those of you who want to go it alone, you will need a 2764 eeprom programmer, a 21 VDC power source, a disassembler such as, Hot Z or another that allows block transfers of memory, and enough memory to hold the disassembler and 8K of machine code (0000 to 1FFF). Another help may be access to an eeprom eraser, in case it's not quite right the first time.

While you are at it, it is time to get a monitor, there are several sources for these. A good 12" green screen composite video (75 ohm) for \$80 to \$90 is a good buy. Since you have a color computer now, a better bet would be a color monitor. Panasonic makes one for about \$225, but there are several better ones. If you ever switch to a 2068 or some lesser computer, then the monitor can be used with it as well. We do not recommend using a TV modulator, because the rf interference caused by the high frequency of board 'A' (>10 MHz), may be intolerable. Shielding would probably help though.

Now, you are ready to 'power-er-up'. Insert boards 'A' and 'B' with Ea installed, on your expansion board. Plug your expansion board into the computer and connect your monitor to board 'A's video jack. With no other boards attached, at all, plug in your new power supply. Good Luck!!

THE FINE PRINT

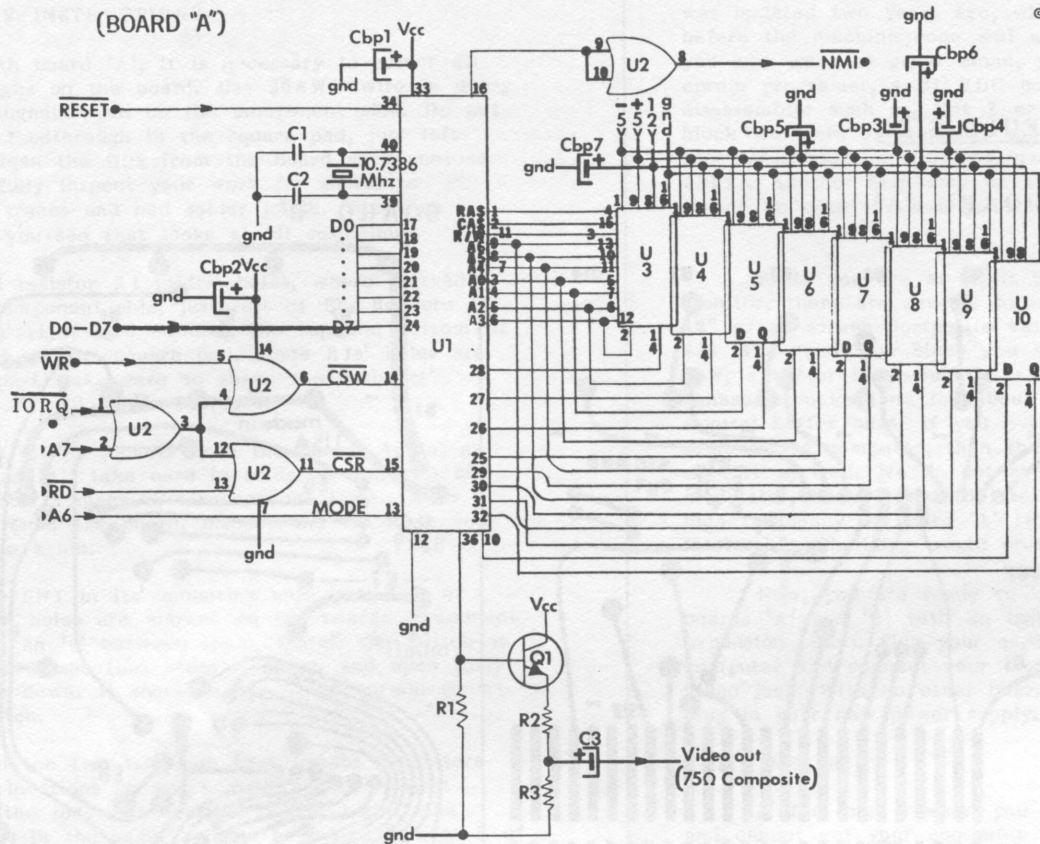
If for some reason, you try all of the tests and cannot get your computer to recover or the cursor does not appear, then regress and start undoing some of your work and see if you can repair it. You can usually find the problem if you look long enough. We at SWN and many other people have this upgrade running (although I sent my board back to John for repair- a short in the 16K memory array), we can't be held responsible for your errors. We will however help you in any way we can. John services what he sells. Send boards 'A' and 'B' (completely assembled), with your eeprom and a check for \$15.00 to him and he will check it out. This will eliminate the video boards from the problem. If your computer still does not work, then try checking the power supply, expansion board connections and any other add on circuitry you have added.

ROM CS not NOT ALLOWED

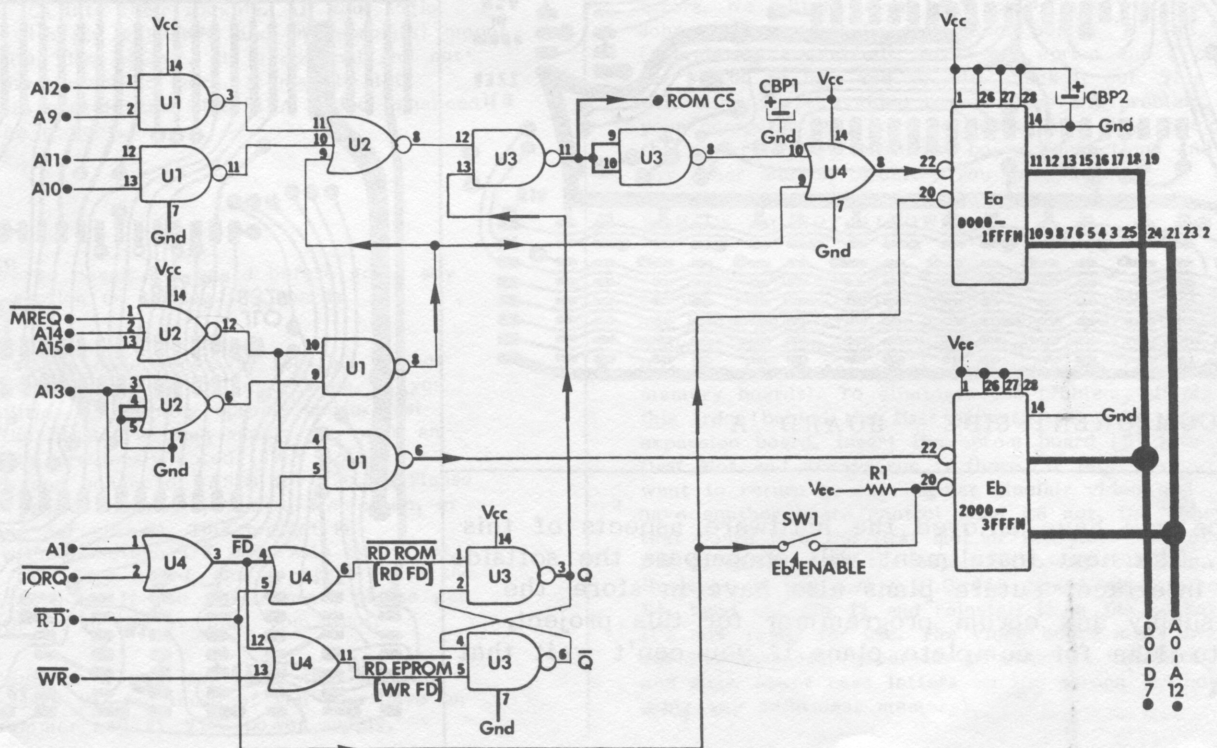
This upgrade will work with several other addons. The one restriction is the ROM chip select not line (trace 23B, the bottom right trace on the back of the computer). Many boards try to control this line (character generators, Memotech I/F, 64K memory boards). To eliminate the problem, simply cut this trace behind the first expansion slot on your expansion board. Insert the eeprom board (B) into the first slot and always run it there. If later you want to return to the regular Sinclair video and have another board control ROM CS not, then plug that board into the first slot or into the back of the computer, with the expansion board following. However, if you own a JLO 64K board, then remove ic, U5, bend out pin 15 and reinstall it in the socket. It's now ready for use. The video board must swap out the character generator in order to initialize and show lower case letters on the screen (without using any additional memory).

(BOARD "A")

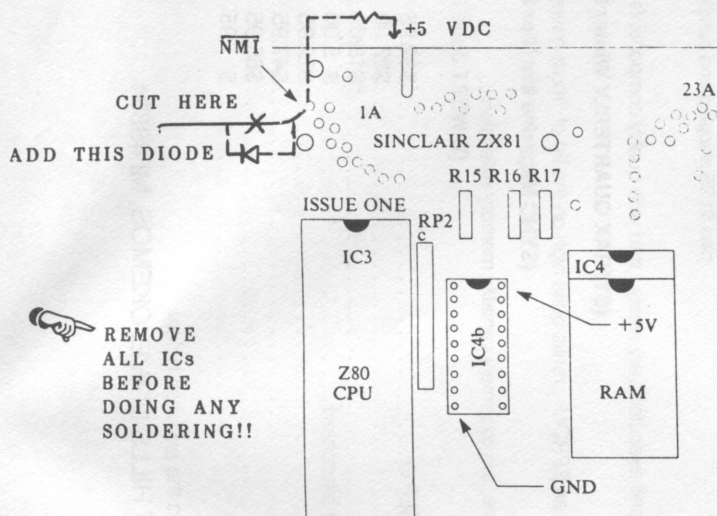
© The John Oliver Co.



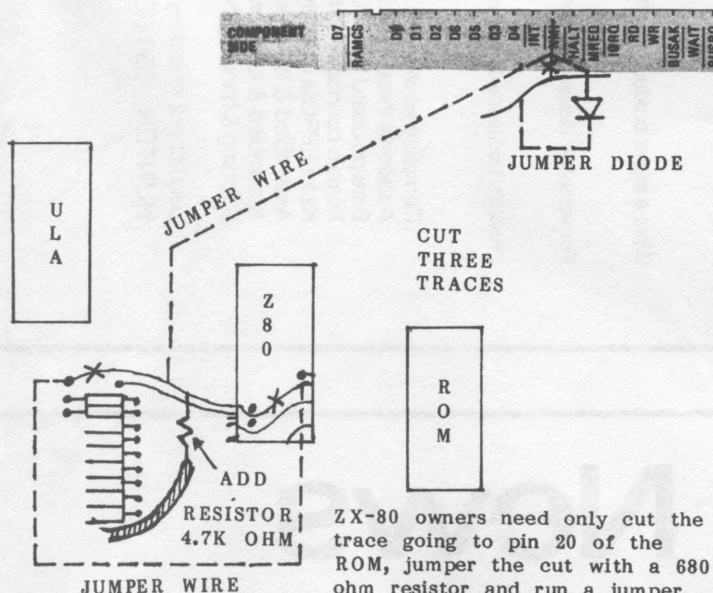
(BOARD "B"-TO BE USED IN CONJUNCTION WITH BOARD "A")



ADD THIS RESISTOR



NEWER TYPE BOARDS



PARTS LIST BOARD B

- U1 74LS00 Quad 2-input NAND gate
- U2 74LS27 Triple 3-input NOR gate
- U4 74LS32 Quad 2-input OR gate
- Ea, Eb(opt.) 2764 eprom 300ns or faster
- R1 10K Ohm 1/4 watt resistor
- Cbp1, Cbp2 12 MF/6V axial tantalum cap.
- SW1 SPDT ultra-sub mini PC mount slide switch(cut 1 lead off)
- MISCELLANEOUS
- 4 14 pin soldertail IC sockets
- 2 28 pin soldertail IC sockets
- 12" 30AWG solid bare wire (for feedthroughs)

Bare board A -Main VDP I/F \$14.95
 Bare board B -Firmware/bank select board - \$11.95
 Both together - \$24.95
 Parts kit A - \$22.95
 Parts kit B - \$5.95
 Parts kits together - \$27.95
 All of the above - \$48.95 (save \$6.95)

TESTING YOUR WORK

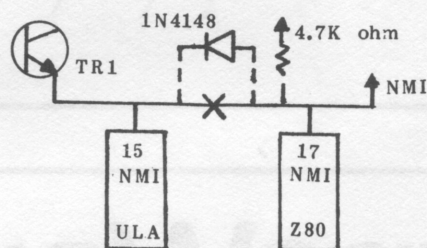
BOARD 'A'

After assembling board 'A', completely, we can now partially (smoke) test it. Remove U2 (74LS32) and bend out pin 8. Reinstall the IC the the socket with pin 8 extending over the side, not entering the socket. Now, insert board 'A' into your expansion board and power up the computer. With the regular TV output from the computer, the cursor should appear as usual. With the monitor connected to the output jack on the video board, you should get a blank screen. If either of these do not occur, then power down and remove your 'A' board and start hunting for your shorts (keep your pants on please) or other errors. The most likely place for a short or solder bridge, is in the memory array (I found three in mine-ed.). Be sure the tantalum bypass caps are installed correctly (especially Cbp 4).

Now remove U2, bend in pin 8 and reinstall the IC in its' socket. Be sure all pins make contact correctly.

BOARD 'B'

The initial test for this board is done by removing any eproms that are installed, and also by removing U3. Insert this board into your expansion board and power up your computer. With the video taken from the normal Sinclair source, you should see the cursor appear on the screen as usual. If not, then power down and start looking for the problem.



SCHEMATIC REPRESENTATION

PARTS LIST BOARD A

- U1 TMS9918A Video Display Processor**
- U2 74LS32 Quad 2-input OR gate
- U3-U10 4116 Dynamic Ram, 150ns.
- Y1 10.7386 Megahertz Crystal**
- Q1 2N3904 NPN transistor
- R1 470 Ohm 1/4 watt resistor (yel/vio/brn)
- R2 30 Ohm 1/4 watt resistor (or/blk/blk)
- R3 75 Ohm 1/4 watt resistor (vio/grn/blk)
- C1, C2 33 Picofarad ceramic capacitor
- C3 50 Microfarad/6V electrolytic capacitor
- Cbp1-Cbp5 12 MF/6V axial tantalum cap.
- Cbp6, Cbp7 10 MF/16V tantalum cap.
- MISCELLANEOUS
- 1 14 pin soldertail IC socket
- 8 16 pin soldertail IC sockets
- 1 40 pin soldertail IC socket
- 1 female RCA (phono) type connector
- 8" coaxial cable
- 2.5" 22AWG strand wire (for jumper)
- 2" 22AWG solid bare wire (for cable clamp)
- 12" 30AWG solid bare wire (for feedthroughs)

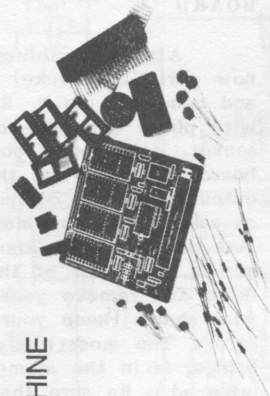
** not included in parts kit.

The HUNTER Board

Add Memory that won't Forget!

DESCRIBED IN JULY/AUGUST 1983 **Radio-Electronics**

- ✓ ADD YOUR OWN SYSTEM UTILITIES
- ✓ BUILD UP A LIBRARY OF MACHINE LANGUAGE SUBROUTINES
- ✓ UP TO 8K NONVOLATILE RAM
- ✓ USE HM6116LP CMOS RAM OR 2716/2732 EPROM
- ✓ COMPATIBLE WITH 16K RAM PACKS



\$3295

plus \$1.95 shipping and handling

What a super product!...conceived and executed very nicely...and with quality components.

(SYNTAX QUARTERLY Winter 82)

For versatility this is even better than an EPROM...ranks quite high on the list of "must-haves" ...

(SYNC Magazine Mar/Apr 83)

Provides the user with instant software...an extremely versatile memory extension...

(Z-WEST June 83)

Complete kit with one 2K 6116LP-3.....	\$32.95
Additional three 6116LP-3.....	\$22.00
Bare pc board & manual.....	\$13.05
Female connector 23/46 gold bifurcated.....	\$ 5.00
Kit for EPROM use only.....	\$22.95
Assembled & tested with 2K.....	\$47.95
Assembled & tested with 8K.....	\$65.95
Shipping & handling per order.....	\$ 1.95

Send check or money order to the address below:

HUNTER, 1630 FOREST HILLS DRIVE, OKEMOS, MI 48864

SyncWare News

Contents

FOR YOUR SUPPORT.....	2
FORUM.....	2
BASIC TOPICS.....	3
Numerical Input, Nachbaur	
2068 Felt Hat, Bingham	
2068 BASIC ROM CALLS.....	4
Kingsley	
FORTH WITH.....	8
Smith	
BASIL'S COMPENDIUM.....	8
Wentworth	
TRANSLATIONS.....	12
Memotext-Pro/File	
Connection, Nachbaur	
A VIDEO UPGRADE.....	14
Oliger	

Published bi-monthly by Thomas B. Woods, P.O. Box 64, Jefferson, NH 03583

TO:

Editor: Tom Bent
9016 Flicker Place
Columbia, MD 21045

Submissions, announcements, letters to the editor, and all other material of editorial interest should be sent to the Maryland address.

Copyright 1984, SyncWare News

Subscribe ! \$ 16.95

Add \$3 a year in Canada; all other foreign add \$5 per year.

1 year (6 issues)